SDC
STORAGE DEVELOPER CONFERENCE
SNIA SANTA CLARA, 2016

Performance Analysis of the Peer Fusion File System (PFFS)

Richard Levy
Peer Fusion

Topics

- Context
- Overview of the Peer Fusion File System
- PFFS Architecture
- Performance
- Getting Started

Context: Hyper growth in DATA is multiplying the cost and risk challenges for ALL organizations

- DATA is the most precious asset of business in the Information age
  - Represents years of labor, great expenditure and must be highly available. Most businesses would shutdown by a catastrophic loss of their data
  - Growth rate increasing exponentially (10x data by 2020)
- COSTS and RISKS associated with storing Big Data are not trivial
  - Replication often requires three copies
  - Three times the cost of hardware, datacenter floor space, power, cooling and administration.
  - And, still the relatively common occurrence of one server out of thousands failing leaves the datacenter with just one spare copy of the content lost!
  - The urgent task of making a third copy can take many hours depending upon the number of terabytes of data lost.

Context: Based on a unique approach and patented technology, a new solution has arrived to address these challenges!

- **Peer Fusion File System** is the solution that reduces these costs and mitigate the risks
  - ensures high resiliency as the data is ingested and without replication
  - Patented technology uses a proprietary protocol (MBP) that ingests data into storage clusters at essentially wire speed.
  - The resiliency of a single storage cluster can be configured to protect against the loss of a few peers to the loss of over 80% of the peers.
- **BOTTOM LINE:**
  The majority of the peers in a cluster can be lost without applications being disrupted AND on-the-fly repairs are performed automatically!

Architecture of the PFFS

- A parallel file system:
  - A Cluster of storage peers
  - Data is striped across the cluster
  - Highly resilient with no user data replication
  - High performance – highly parallel
  - The workload is balanced across the cluster
  - Simple to Administer

Architecture: Resiliency

- Resiliency is ensured through a Reed-Solomon (RS) erasure codec.
- A RS codeword consists of one 8KB block per peer.
- Some blocks are user data and others checksum data.
- Forward Error Correction (FEC) is user configurable.
- FEC ranges from 1 peer to 90% of the cluster.
- Data regeneration is performed equally by all peers.

Architecture: Parallelism

- Most intra-cluster network communication is through UDP/IP multicast.
- All the cluster peers receive commands and data in parallel.
- Peers respond to commands in parallel:
  - Store/xmit data (disk/network I/O)
  - Create/delete file, etc.
- Threads dedicated to draining each NIC and queuing messages.
- Thread pools for message triage.
- Threads dedicated to processing session messages.

Architecture: The Directory Cache

- The PFFS namespace is replicated across the cluster for resiliency.
- Gateways create a local Directory Cache (DC) to avoid cluster queries `(opendir,readdir,readlink,stat, etc.)`
- The DC is not persistent – it is wiped on Gateway start-up.
- On `opendir` Gateways read the directory from the cluster and populate the DC. Subsequently only the local caches are consulted for that directory.
- The DC is updated by commands `(create,link,symlink, unlink,chown,chmod, mkdir,rmdir, etc.)`

Architecture: Cluster Healing

- The PFFS is designed to continue operating even with peer failures.
- Healing is necessary for failed peers that return to service.
- Healing walks the PFFS tree ascertaining the health of each directory entry by collecting status from the peers to achieve a quorum.
- Only directory entries whose peers are not part of the quorum require healing.
- All failed peers are healed simultaneously with a small performance cost for each additional peer.

Architecture: The Multicast Burst Protocol

- A many-to-many protocol for the reliable and efficient delivery of multicast messages.
- A Burst is an ordered group of related messages that must be processed together.
- MBP message headers completely define the burst context.
  - Some peers can miss an original Burst command, yet infer the full context through the replies of other peers.
  - Intra-cluster chatter is minimized as each peer can automatically compute exactly what all the other peers will do (e.g. store user data, compute checksum data, repair data, etc.)

Architecture: Roadmap

- We plan to investigate:
  - Hardware assisted Reed Solomon codecs.
  - New NIC drivers that eliminate interrupts and decrease latency.
  - Infiniband to improve on Ethernet latency.
  - Develop a client-side PFFS
- We plan to obtain additional metrics:
  - iometer, iozone, bonnie++, etc.
- We just filed a patent for a 64k-peer cluster.

Performance: Node Configurations

- Peers and gateway:
  - Hardware:
    - CPU: Quad-core i5-2400@3.10GHz
    - RAM: 8GB
    - HDD: SATA2 500GB@7200RPM
    - NICs: 1 to 6 x1000Mb/s
  - Software:
    - CentOS 7.0 (generic kernel)
    - Default NIC configurations (no tuning)

Performance: Cluster Configurations

- Resiliency:
  - FEC code rate is computed as min_peers/max_peers. It represents the percentage of useful information in a codeword. This is a measure of how efficiently the storage capacity Is used.
  - The repair rate computed as failure_count/max_peers represents the percentage of information to be regenerated in a codeword.
  - For this presentation we tested:
    - FEC settings from 10% to 88%.
    - Repair rates from 11% to 77%.
    - Cluster sizes from 3 peers to 20 peers.
- Tests were run with the Cluster mounted in direct I/O and buffered I/O modes.

Performance: Configuration Files

- Gateway configuration file:

```
BURST_BUFS=20000     ←Pre-allocated burst buffers
IOBUFS=131072        ←Data staging throttle
MAX_PEERS=20
MIN_PEERS=2
CLI_THREADS=10       ←Namespace commands thread pool
CLUSTER_LAN=(enp1s0f0(6000), enp1s0f1(6002),
             enp2s0f0(6004), enp2s0f1(6006),
             enp3s0f0(6008), enp3s0f1(6010))
ROOT_MCAST_ADDR=225.100.100.100  ←Cluster id
DIR_CACHE="/home/pf0/PFDirCache"
```


SDC 16

Performance: Configuration Files

- Peer configuration file:

```
CLI_THREADS=10
CLUSTER_LAN=(enp1s0f0(6000), enp1s0f1(6002),
             enp2s0f0(6004), enp2s0f1(6006),
             enp3s0f0(6008), enp3s0f1(6010))
ROOT_MCAST_ADDR=225.100.100.100
ROOT_DIR="/home/pf0/pffsroot”
```

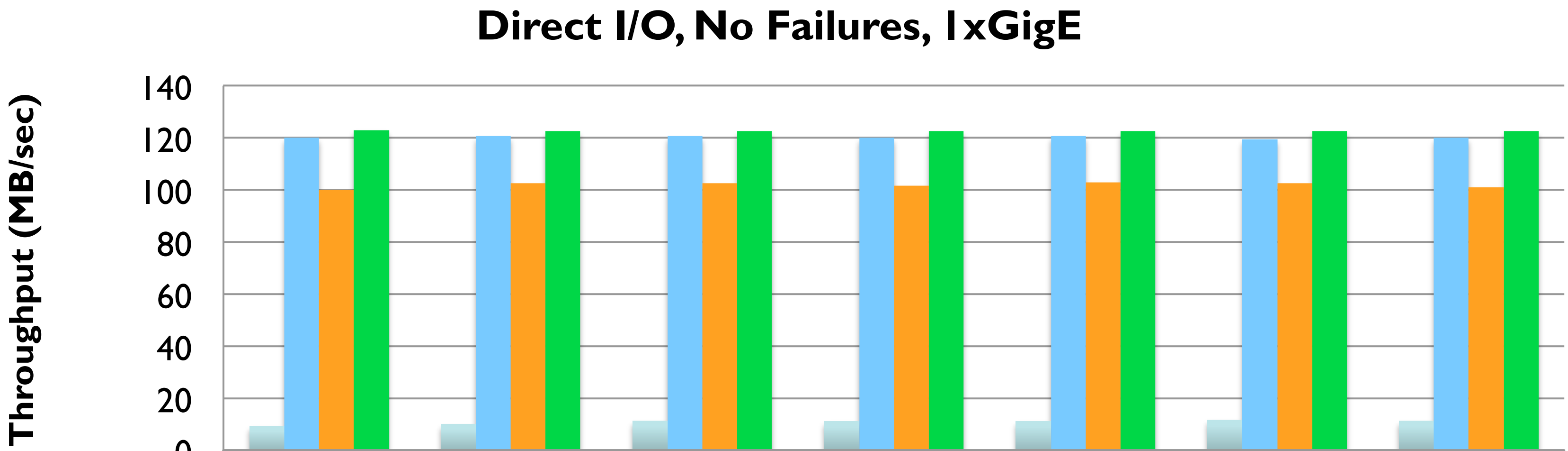
Performance: Considerations

- In general better hardware helps achieve better performance.
- Our metrics were obtained with relatively low-end equipment for better visibility and analysis.
- We can get better performance with better hardware.
- We wanted to test the throughput of a PFFS cluster within the cluster LANs.
- We did not want to measure client LAN performance or client computer I/O performance.

Performance: The tests

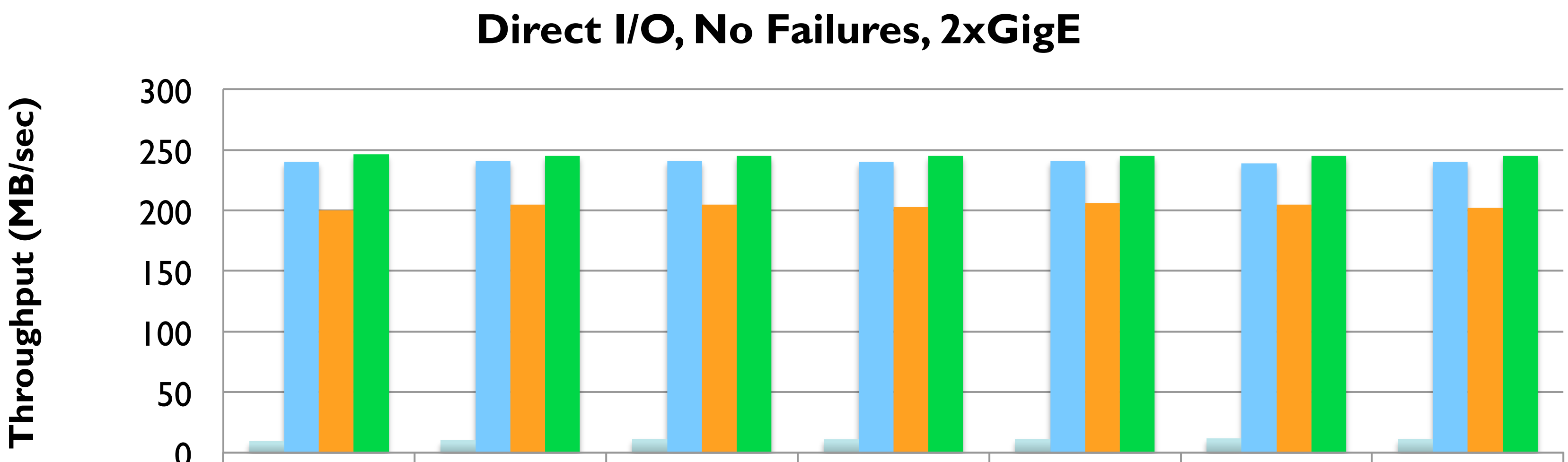
- The Gateway wrote and read all data to avoid client LAN limitations.
- The Gateway performed no disk access for the tests.
- All I/O was sequential, writing was always at EOF.
- The Linux `dd` utility was used:
  - `dd if=/dev/zero of=/pf0/b.dat bs=4k count=262144`
  - `dd if=/pf0/b.dat of=/dev/zero bs=4k count=262144`
  - `dd if=/dev/zero of=/pf0/b.dat bs=1M count=1024`
  - `dd if=/pf0/b.dat of=/dev/zero bs=1M count=1024`

PFFS Performance



	28% FEC	33% FEC	40% FEC	50% FEC	60% FEC	75% FEC	85% FEC
Read 4KB	9.2	9.9	11.2	11	11.1	11.6	11.2
Write 4KB	120	120.5	120.5	120	120.5	119.5	120
Read 1MB	100	102.5	102.5	101.5	103	102.5	101
Write 1MB	123	122.5	122.5	122.5	122.5	122.5	122.5

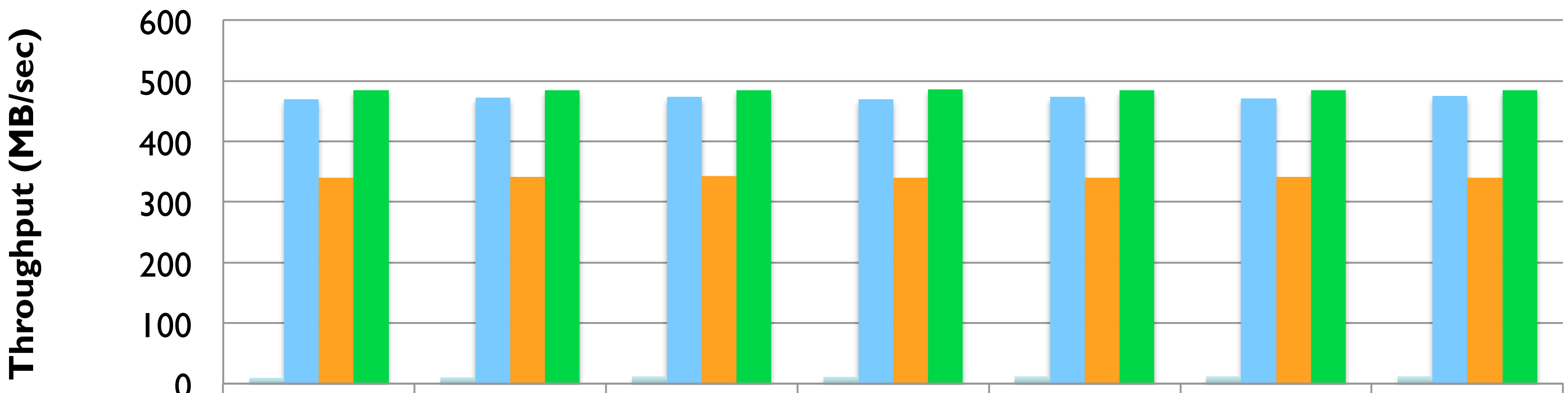
PFFS Performance



	28% FEC	33% FEC	40% FEC	50% FEC	60% FEC	75% FEC	85% FEC
Read 4KB	9.2	9.9	11.2	11	11.1	11.6	11.2
Write 4KB	240	241	241	240	241	239	240
Read 1MB	200	205	205	203	206	205	202
Write 1MB	246	245	245	245	245	245	245

PFFS Performance

Direct I/O, No Failures, 4xGigE

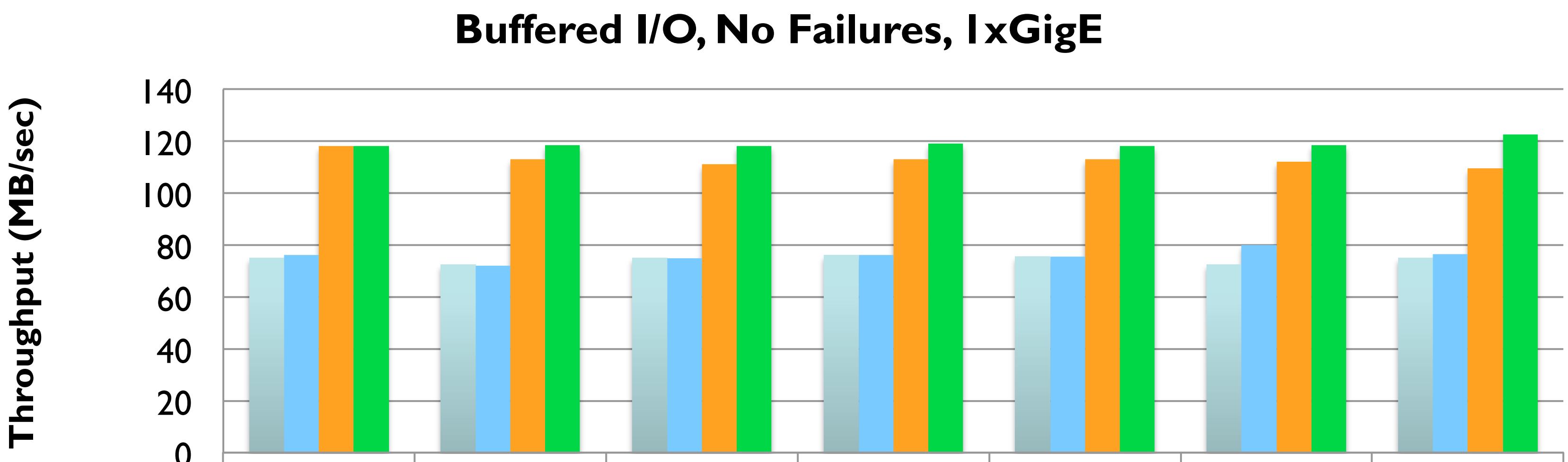


	28% FEC	33% FEC	40% FEC	50% FEC	60% FEC	75% FEC	85% FEC
Read 4KB	9.2	9.9	11.2	11	11.1	11.6	11.2
Write 4KB	470	472	473	470	473	471	475
Read 1MB	340	341	342	340	340	341	340
Write 1MB	484	485	485	486	485	485	485

PFFS Performance

	28% FEC	33% FEC	40% FEC	50% FEC	60% FEC	75% FEC	85% FEC
Read 4KB	9.2	9.9	11.2	11	11.1	11.6	11.2
Write 4KB	656.2	653	655	654	656	656	655
Read 1MB	417	420	424	414	421	413	428
Write 1MB	729	727	729	728	730	728	729

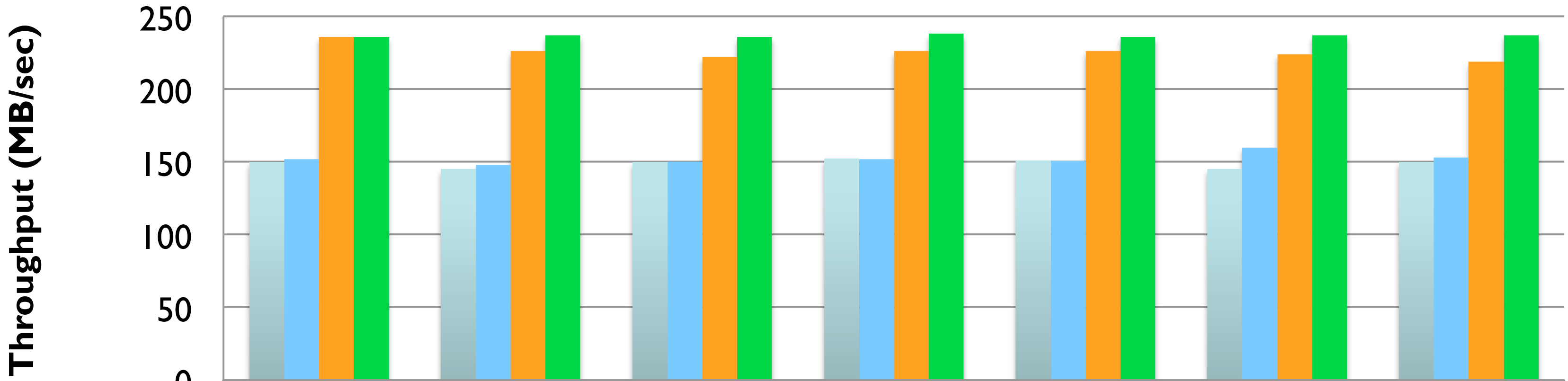
PFFS Performance



	28% FEC	33% FEC	40% FEC	50% FEC	60% FEC	75% FEC	85% FEC
Read 4KB	75	72.5	75	76	75.5	72.5	75
Write 4KB	76	72	75	76	75.5	80	76.5
Read 1MB	118	113	111	113	113	112	109.5
Write 1MB	118	118.5	118	119	118	118.5	122.5

PFFS Performance

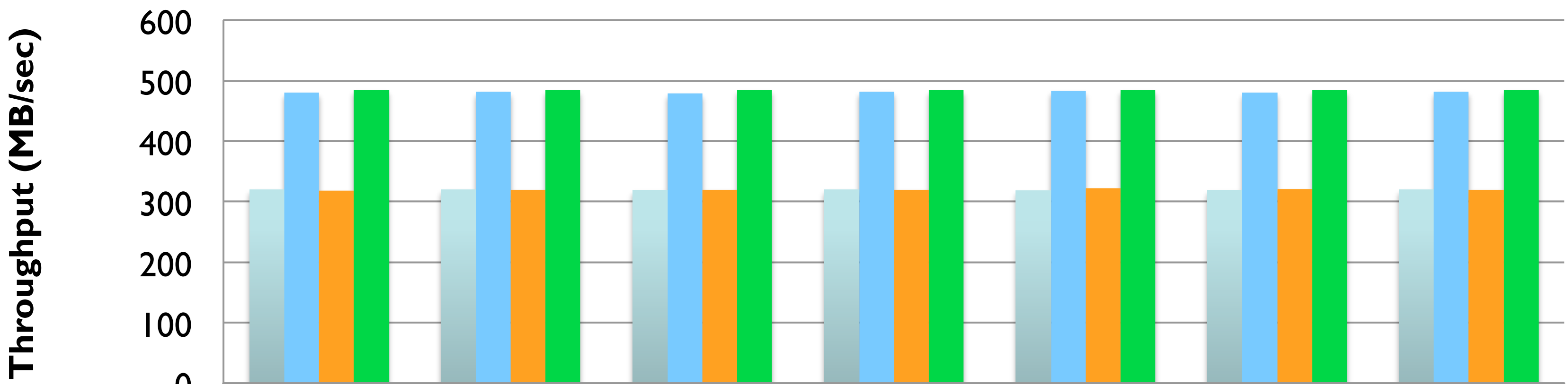
Buffered I/O, No Failures, 2xGigE



	28% FEC	33% FEC	40% FEC	50% FEC	60% FEC	75% FEC	85% FEC
Read 4KB	150	145	150	152	151	145	150
Write 4KB	152	148	150	152	151	160	153
Read 1MB	236	226	222	226	226	224	219
Write 1MB	236	237	236	238	236	237	237

PFFS Performance

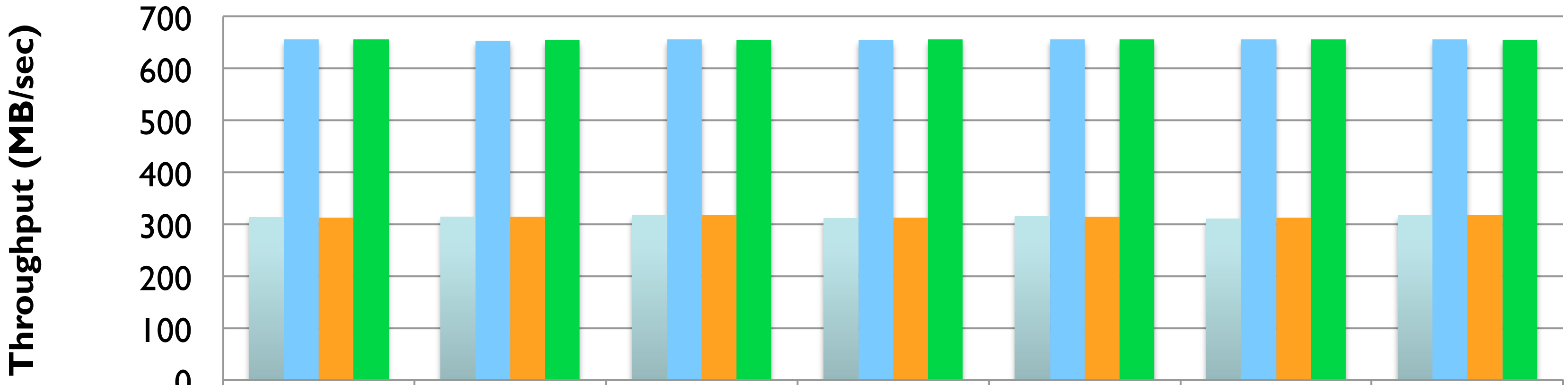
Buffered I/O, No Failures, 4xGigE



	28% FEC	33% FEC	40% FEC	50% FEC	60% FEC	75% FEC	85% FEC
Read 4KB	320	320	319	320	318	319	320
Write 4KB	480	481	479	482	483	480	482
Read 1MB	318	319	320	320	322	321	320
Write 1MB	485	484	485	484	485	484	485

PFFS Performance

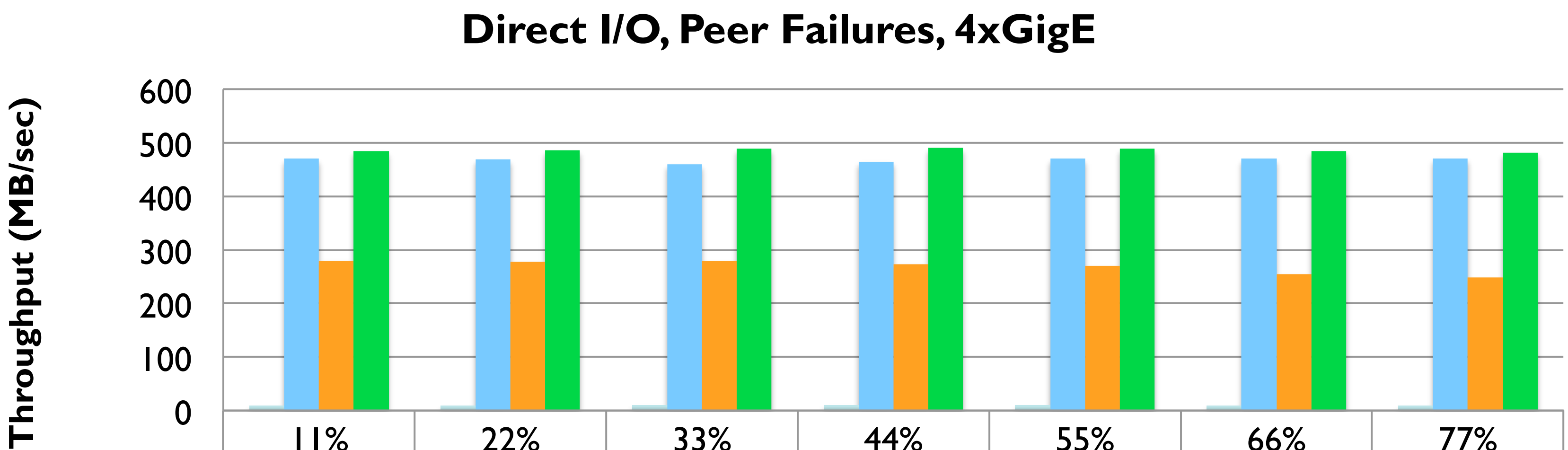
Buffered I/O, No Failures, 6xGigE



	28% FEC	33% FEC	40% FEC	50% FEC	60% FEC	75% FEC	85% FEC
Read 4KB	313	314	318	311	315	310	317
Write 4KB	656.2	653	655	654	656	656	655
Read 1MB	313	314	317	312	314	312	317
Write 1MB	656	654	654	655	656	655	654

SDC 16

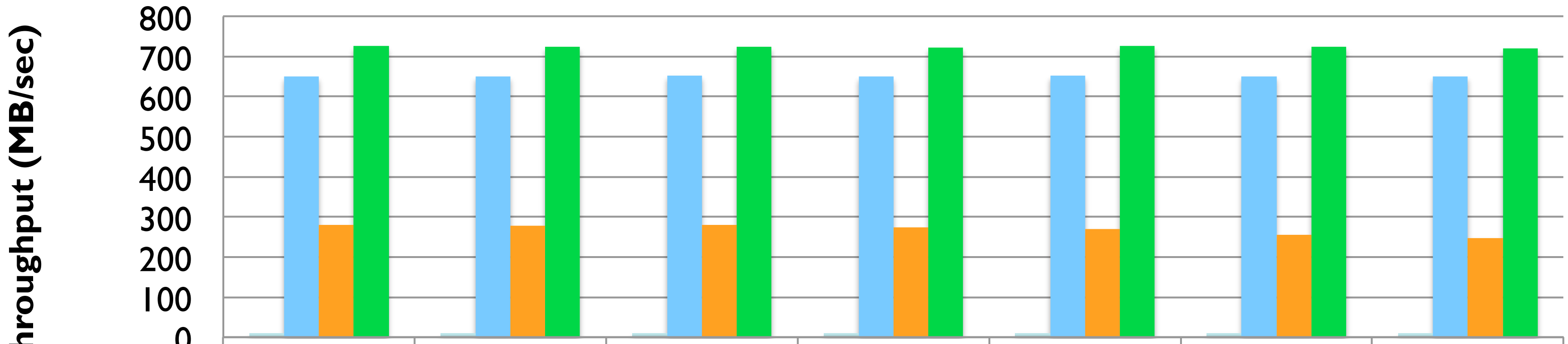
PFFS Performance With Peer Failures



	11% Repairs	22% Repairs	33% Repairs	44% Repairs	55% Repairs	66% Repairs	77% Repairs
Read 4KB	9	9.1	9.9	9.7	9.4	9.3	9.1
Write 4KB	470	469	460	465	470	470	471
Read 1MB	280	278	279	274	270	255	248
Write 1MB	485	486	489	490	489	485	482

PFFS Performance With Peer Failures

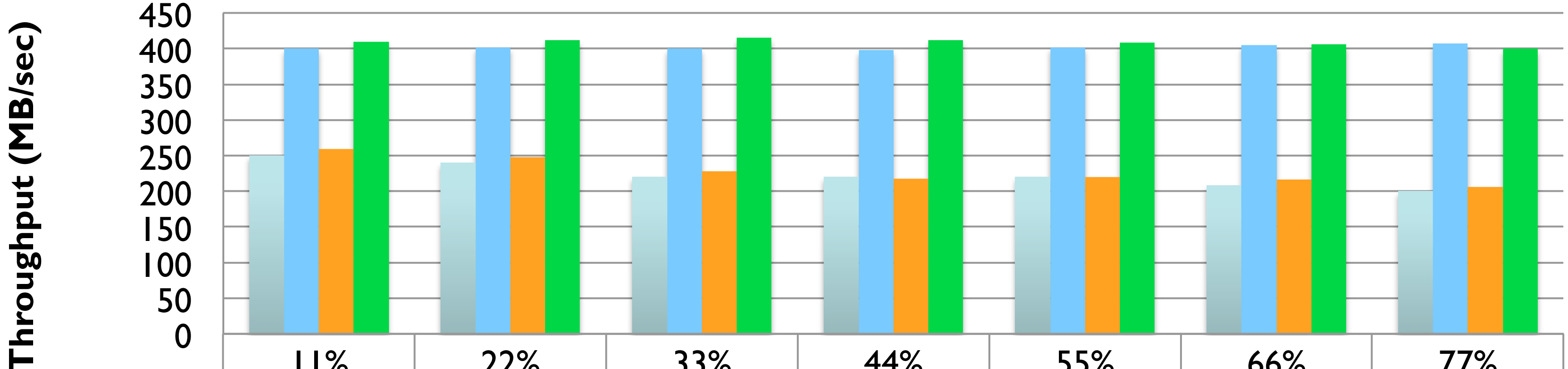
Direct I/O, Peer Failures, 6xGigE



	11% Repairs	22% Repairs	33% Repairs	44% Repairs	55% Repairs	66% Repairs	77% Repairs
Read 4KB	9	9.1	9.9	9.7	9.4	9.3	9.1
Write 4KB	650	651	653	650	652	651	651
Read 1MB	280	278	279	274	270	255	248
Write 1MB	726	724	725	723	727	725	720

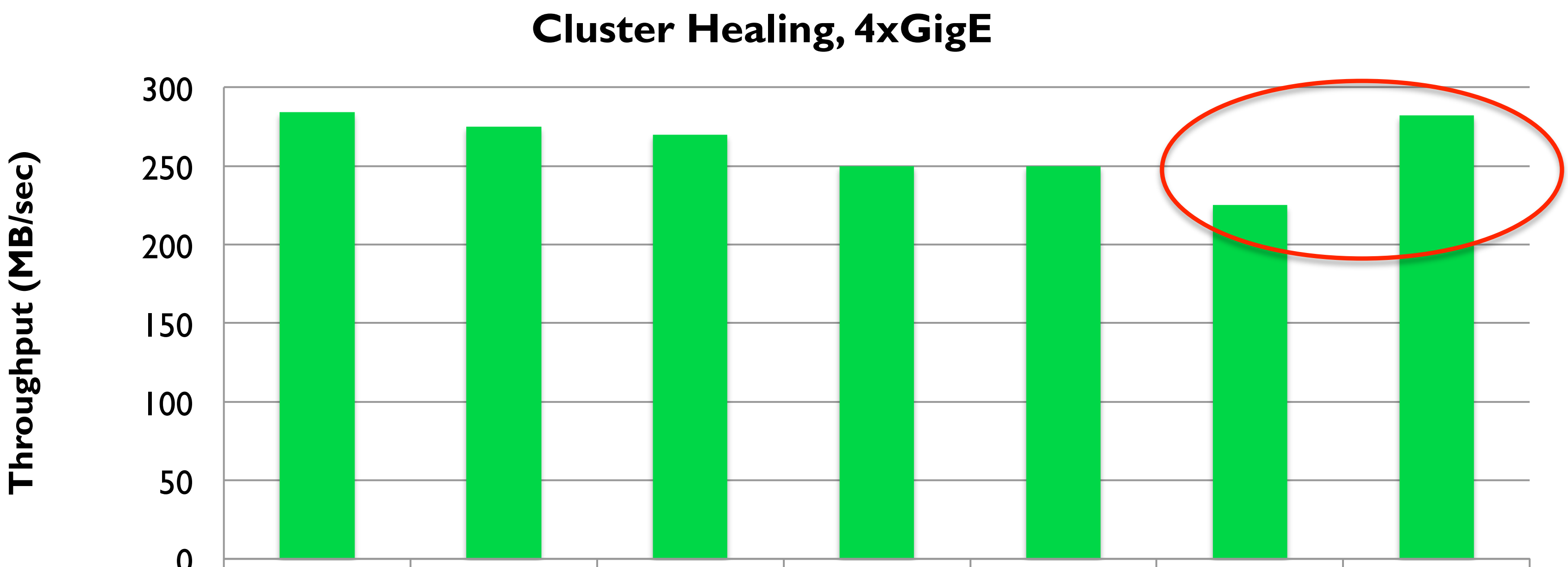
PFFS Performance With Peer Failures

Buffered I/O, Peer Failures, 4xGigE



	11% Repairs	22% Repairs	33% Repairs	44% Repairs	55% Repairs	66% Repairs	77% Repairs
Read 4KB	250	240	220	220	220	208	200
Write 4KB	400	402	400	398	402	405	407
Read 1MB	259	248	228	218	220	217	206
Write 1MB	410	412	415	412	409	406	400

PFFS Performance - Healing



	20% Repairs	25% Repairs	33% Repairs	40% Repairs	50% Repairs	60% Repairs	77% Repairs
Repair Rate	284	275	270	250	250	225	282

Performance: Observations I

- In general better hardware helps. Our metrics were obtained with low-end equipment for better visibility and easier analysis.
- Network bandwidth is very important for performance.
- Under nominal conditions the CPU was under 10%.
- For heavy repair loads the CPU reached 20%.
- There was no memory pressure on the peers.
- The performance of the peers' single HDD was easily maxed-out in clusters with low peer counts and high NIC counts.

Performance: Observations II

- Buffered I/O is vastly better for small I/O.
- Direct I/O is significantly better for large I/O.
- Performance degrades gracefully (linearly) as peer failures are injected.
- Appending to a file is essentially wire-speed in direct I/O mode regardless of the failure rate.
- Healing peer failures performance amounts to a small penalty on the corresponding reading 1MB in direct I/O mode.

Getting Started

- Visit www.peerfusion.com and look for the SDC link.
- Get more detailed information on PFFS.
- SDC attendees can download a white paper.
- Apply for a very limited beta program through 10/15/2016

Thank You!

Richard Levy
Peer Fusion, Inc.
Founder/CEO
richard@peerfusion.com